Power and Performance Characterization and Modeling of GPU-Accelerated Systems

Yuki Abe
Kyushu University
Fukuoka, Japan
abe@soc.ait.kyushu-u.ac.jp

Hiroshi Sasaki
Kyushu University
Fukuoka, Japan
sasaki@soc.ait.kyushu-u.ac.jp

Shinpei Kato
Nagoya University
Nagoya, Japan
shinpei@is.nagoya-u.ac.jp

Koji Inoue
Kyushu University
Fukuoka, Japan
inoue@ait.kyushu-u.ac.jp

Masato Edahiro
Nagoya University
Nagoya, Japan
eda@is.nagoya-u.ac.jp

Martin Peres
Laboratoire Bordelais de Recherche en Informatique
Bordeaux, France
martin.peres@labri.fr

***Abstract*—Graphics processing units (GPUs) provide an order-of-magnitude improvement on peak performance and performance-per-watt as compared to traditional multicore CPUs. However, GPU-accelerated systems currently lack a generalized method of power and performance prediction, which prevents system designers from an ultimate goal of dynamic power and performance optimization. This is due to the fact that their power and performance characteristics are not well captured across architectures, and as a result, existing power and performance modeling approaches are only available for a limited range of particular GPUs. In this paper, we present power and performance characterization and modeling of GPU-accelerated systems across multiple generations of architectures. Characterization and modeling both play a vital role in optimization and prediction of GPU-accelerated systems. We quantify the impact of voltage and frequency scaling on each architecture with a particularly intriguing result that a cutting-edge *Kepler*-based GPU achieves energy saving of 75% by lowering GPU clocks in the best scenario, while *Fermi*- and *Tesla*-based GPUs achieve no greater than 40% and 13%, respectively. Considering these characteristics, we provide statistical power and performance modeling of GPU-accelerated systems simplified enough to be applicable for multiple generations of architectures. One of our findings is that even simplified statistical models are able to predict power and performance of cutting-edge GPUs within errors of 20% to 30% for any set of voltage and frequency pair.**

***Keywords*-GPUs, power, performance, characterization, modeling.**

I. Introduction

Graphics processing units (GPUs) are increasingly used as a means of massively parallel computing. Even consumer series of GPUs integrate more than $1,500$ processing cores on a single chip and the peak double-precision performance exceeds 1 TFLOPS while sustaining thermal design power (TDP) in the same order of magnitude as traditional multicore CPUs [21]. Current main applications of general-purpose computing on GPUs (GPGPU) can be found in supercomputing [26] but there are more and more emerging applications in other areas. Examples include autonomous vehicles [17], software routers [6], encrypted networks [11] and storage systems [13, 25].

One of the grand challenges of GPU-accelerated systems is power and performance optimization. GPUs bring significant advantages in "peak" performance and performance-per-watt, but if some functional units are left unused during operation, GPUs can waste non-trivial power and computing resources. Voltage and frequency scaling may address this problem. However, there is little work for how to predict power and performance of GPU-accelerated systems based on given voltage and frequency pair.

Problem Definition: Power and performance prediction requires characterization and modeling of corresponding systems. Previous work on power and performance characterization and/or modeling [7, 8, 12, 14]–[16, 18, 23, 27] restricted their attention to specific GPUs, and therefore it is questionable if their results can be applied for other GPUs. Most of these work also focus on processing core performance, though they desire to be complemented by modeling and scaling of memory performance. In order to develop a generalized method of power and performance prediction for GPU-accelerated systems, characterization and modeling must be performed on multiple generations of architectures.

Contribution: This paper presents power and performance characterization and modeling of GPU-accelerated systems based on multiple generations of architectures. We first characterize the architectural impact of voltage and frequency scaling on execution time and power consumption, followed by providing statistical power and performance modeling of GPU-accelerated systems. These characterization and modeling both play a vital role in prediction and optimization of power and performance. To the best of our knowledge, this paper is the first scientific evidence beyond vendor's specification sheets that successfully characterizes power and performance of GPU-accelerated systems across multiple generations of architectures, and derives a unified form of statistical model to predict their power and performance.

Organization: The rest of this paper is organized as follows. Section II presents the platform, upon which we explore power and performance characterization and modeling. Section III characterizes impact of voltage and frequency scaling on minimized energy of GPU-accelerated systems using four different GPUs. Section IV provides statistical modeling of their power and performance. Related work are discussed in

TABLE I
SPECIFICATIONS OF THE NVIDIA GPUS.

GPU	GTX 285	GTX 460	GTX 480	GTX 680
Architecture	Tesla	Fermi	Fermi	Kepler
# of processing cores	240	336	480	1536
Peak performance (GFLOPS)	933	907	1350	3090
Memory bandwidth (GB/sec)	159.0	115.2	177.0	192.2
TDP (Watt)	183	160	250	195
Core frequency (MHz)	600, 800, 1296	100, 810, 1350	100, 810, 1400	648, 1080, 1411
Memory frequency (MHz)	100, 300, 1284	135, 324, 1800	135, 324, 1848	324, 810, 3004

Section V. We provide concluding remarks of this paper in Section VI.

II. PLATFORM

We use four different NVIDIA graphics cards (GeForce GTX 285, 460, 480 and 680) with an Intel Core i5 2400 processor. The operating system is Linux kernel v3.3.0. GPU programs are all written in the Compute Unified Device Architecture (CUDA) programming language [22].

A. GPU Architecture

This paper focuses on the NVIDIA GPU architectures: *Tesla*, *Fermi* and *Kepler* [19, 21]. Among these different generations of architectures, we explore four representative GPUs of the GeForce consumer series: (i) Tesla-based GTX 285, (ii) Fermi-based GTX 460, (iii) Fermi-based GTX 480 and (iv) Kepler-based GTX 680. There are two Fermi-based GPUs selected because we aim to characterize a difference within the same architecture as well as that among different architectures.

TABLE I illustrates specifications of the four GPUs. They are diverse in the number of processing cores, peak performance, memory bandwidth, TDP and operating frequency. In the rest of this section, we provide an overview of the GPU architectures in consideration.

Tesla: The Tesla architecture is an initial form of GPU technology designed for unified graphics and parallel computing. It introduces the basic functionality of CUDA such as unified shaders, multithreading, shared memory, and barrier synchronization. However it lacks hierarchical caches and there is some architectural limit on the memory size, ECC support, double-precision performance and so on. We use this architecture to observe how GPUs have been improved in power and performance through the Fermi and the Kepler architectures.

Fermi: The Fermi architecture is a significant leap forward in GPU architecture. It overcomes drawbacks of the Tesla architecture while succeeding the architectural baseline. Notable advances with respect to power and performance include hierarchical caches and a larger number of processing cores. Comparing the Fermi and the Tesla architectures, one can discuss implication of the presence of L1/L2 caches and the scalability of processing cores.

Kepler: The Kepler architecture is the current (as of 2013) state of the art of NVIDIA GPU technology. This is an enhanced and generalized version of the Fermi architecture while introducing more efficient multithreading support. From application points of view, a Kepler GPU looks like an impressively giant Fermi GPU. This engineering innovation brought significant improvements in performance-per-watt, as shown in TABLE I.

B. System Software

We use NVIDIA's proprietary software [20] including the device driver, runtime library and compiler. Since this software package does not provide a system interface to scale power and performance (voltage and frequency) levels of GPUs, we modify the BIOS image of the target GPU, which is embedded in the device driver's binary code, forcing the GPU to be booted at the specified power and performance levels. This method allows us to choose a pre-defined configurable set of the GPU core and memory clocks listed in TABLE I where voltage is implicitly adjusted with frequency changes. Interested readers for this open method are encouraged to visit the software repository of Gdev [13] and find documentations about voltage and frequency scaling of NVIDIA GPUs.

C. Experimental Equipment

We use the Yokogawa Electric Corporation's WT1600 digital power meter [3] for the measurement of power consumption. This instrument obtains voltage and electric current every $50ms$ from the power outlet of the machine. Power consumption is calculated by multiplying the voltage and current, whereas energy consumption is derived by accumulation of power consumption. The measurement is focused on power and energy of the entire system but not on those of individual CPUs and GPUs because our focus is on the system-wide study. There are also equipment constraints that prevent us from measuring power consumption of the GPU alone as its power is directly supplied from the power supply and also from the PCI bus.

D. Target Workload

The benchmarks used in our experiments include Rodinia [2] and Parboil [24] which are popular benchmark suites for GPGPU studies. We execute each benchmark program with the maximum feasible input data size. We also use the CUDA SDK code samples [22] and basic matrix operation programs with large input data size. TABLE II lists all the benchmarks. For such a program that has an execution time less than $500ms$, we modify the code to repeat the computing kernel of the program until the execution time reaches $500ms$ in order to obtain at least 10 sample points given that the minimum time range of the power meter we use in this paper is $50ms$. All the test programs are compiled using the NVIDIA CUDA Compiler (NVCC) v4.2 [20].

TABLE II
LIST OF BENCHMARKS

Suite	Application name
Rodinia	backprop, bfs, cfd, gaussian, heartwall, hotspot kmeans, lavaMD, leukocyte, mummergpu, lud, nn, nw particlefilter_float, pathfinder srad_v1, srad_v2, streamcluster
Parboil	cutcp, histo, lbm, mri-gridding mri-q, sad, sgemm, spmv, stencil, tpacf
CUDA SDK	binomialOptions, BlackScholes, concurrentKernels histogram64, histogram256, MersenneTwister
Matrix	MAdd, MMul, MTranspose

TABLE III
CONFIGURABLE FREQUENCY COMBINATIONS.

	GTX 285	GTX 460	GTX 480	GTX 680
Core-H, Mem-H	√	√	√	√
Core-H, Mem-M	√	√	√	√
Core-H, Mem-L	√	√	√	√
Core-M, Mem-H	√	√	√	√
Core-M, Mem-M	√	√	√	√
Core-M, Mem-L	√	√	√	√
Core-L, Mem-H	√	–	–	√
Core-L, Mem-M	√	–	–	–
Core-L, Mem-L	–	√	√	–

III. POWER AND PERFORMANCE CHARACTERIZATION

In this section, we characterize power and performance behavior of GPU-accelerated systems to play a vital role in optimization and prediction of GPU-accelerated systems. This characterization is essential to derive power and performance models. We perform an extensive set of experiments using the benchmarks presented in Section II-D. The processing core and memory clocks of GPUs are scaled individually according to the configurable frequency combinations predefined by the NVIDIA specifications, which are summarized in TABLE III. These combinations are abbreviated using the following notation: Core/Mem-H, -M, and -L denote the high, medium and low frequencies of the processing core and memory, respectively, corresponding to TABLE I. For example, (Core-H, Mem-L) of GTX 285 sets the core and memory frequencies to 1296MHz and 100MHz, respectively. Note that frequency pairs depend on each GPU.

The goal of this section is to derive appropriate frequency pairs for each benchmark and GPU that minimize energy. Prior work [12] showed that the total energy consumption of the system can be reduced by slowing down the processing core frequency for memory-intensive workloads or vice versa, which is clearly the most intuitive policy of selecting the frequency pairs. A similar approach is studied well for CPUs where the processor frequency is scaled down for memory-intensive workloads by dynamically capturing the last level cache miss ratio [9, 10].

We first conduct a similar study to check whether we can observe the same tendency among the three generations of GPUs. Due to space constraints, we herein present only the selected results of benchmarking. Fig. 1 shows the performance and the power efficiency, i.e., reciprocal of the energy consumption of `Backprop` which we choose to showcase as an example of compute-intensive workload. The x-axis of each figure represents the processing core frequency and different lines in the figure correspond to different memory frequencies. It clearly explains that this benchmark is compute-intensive for all the generations of GPUs. The power grows linearly according to the x-axis while the performance remains constant across each line. The best power efficiency for GTX 285, GTX 460 and GTX 480 is achieved by (Core-H, Mem-L), while that of GTX 680 is provided with (Core-M, Mem-L). The improvements compared to (Core-H, Mem-H), which is the default setting, in power efficiency are 13%, 39%, 40% and 75% for GTX 285, GTX 460, GTX 480 and GTX 680, respectively, where the performance losses are limited to 2%, 2%, 0.1% and 30%, respectively. Lessons learned from this evaluation are that (i) energy consumption can be successfully reduced while retaining little performance degradation on state-of-the-art GPUs and (ii) even if a program is compute-intensive, minimized energy is not obtained by simply lowering the memory frequency to the lowest level.

We next focus on memory-intensive workload and investigate if power efficiency can be improved with minimal performance degradation like `Backprop`. Fig. 2 shows the performance and the power efficiency of `Streamcluster` which is one of the most memory-intensive benchmarks used in this paper. Albeit memory-intensive workload, performance improves when increasing the processing core frequency for the "Mem-H" configurations. On the other hand, the "Mem-M" and "Mem-L" exhibit constant performance regardless of the processing core frequency. Because of this nature, power efficiency cannot necessarily be improved by reducing the processing core frequency when the memory frequency is set to "Mem-M" or "Mem-L". It is interesting to see that GTX 680 can improve power efficiency by 4.7% with performance degradation of 8.7% using the (Core-M, Mem-H) pair. This means that lowering the processing core frequency for memory-intensive workload is effective to reduce energy on GTX 680.

So far we have studied power and performance of heavily compute- or memory-intensive workload. However other benchmarks likely show more complicated power and performance characteristics with respect to the processing core and memory frequencies. For example, such behavior can be seen in `Gaussian` as shown in Fig. 3. It behaves as compute-bound and memory-bound at different frequency levels and for different GPUs. Even if we compare the same-generation GPUs, i.e., GTX 460 and GTX 480, the power efficiency characteristics differ such that the best configuration is not identical between them, which implies that it is not straightforward to predict an optimal core and memory frequency pair. This observation encourages and raises the need of work on modeling the power and performance of GPU-accelerated systems. We present our statistical modeling approach in Section IV.

We now provide an insight of how the power and performance differ among different generations of GPUs and discuss the growing importance of cooperatively choosing the processing core and memory frequencies. TABLE IV summarizes the configurations achieving the best power efficiency for each benchmark and GPU. Note that it includes all the benchmarking results while most of them are excluded in Figures 1 through 3. The abbreviation in the parenthesis denote the best performing processing core and memory frequency

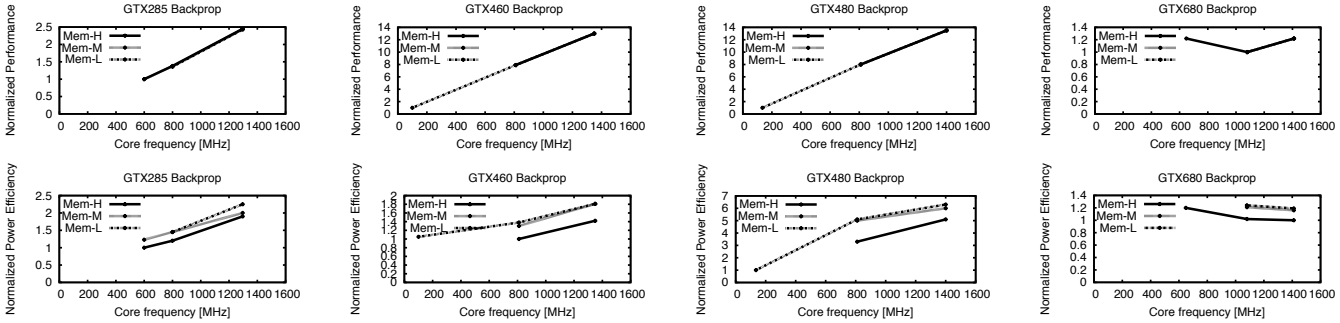

Fig. 1. Performance and power efficiency of `Backprop`.

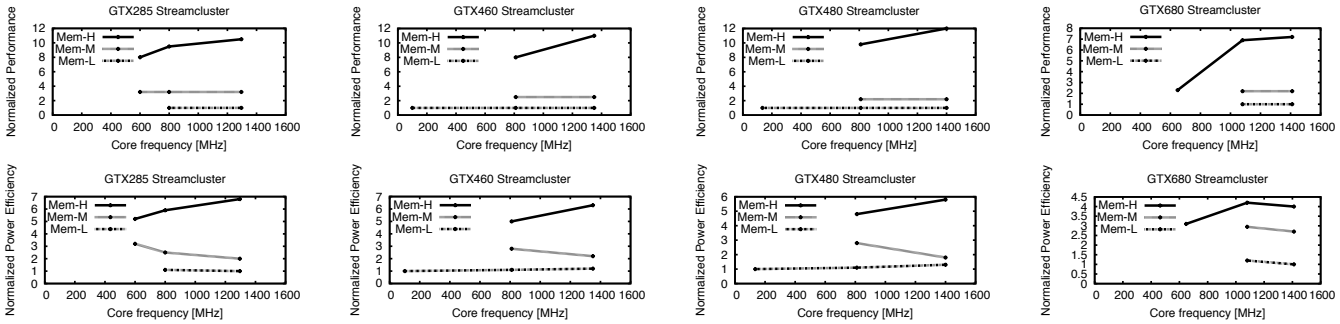

Fig. 2. Performance and power efficiency of `Streamcluster`.

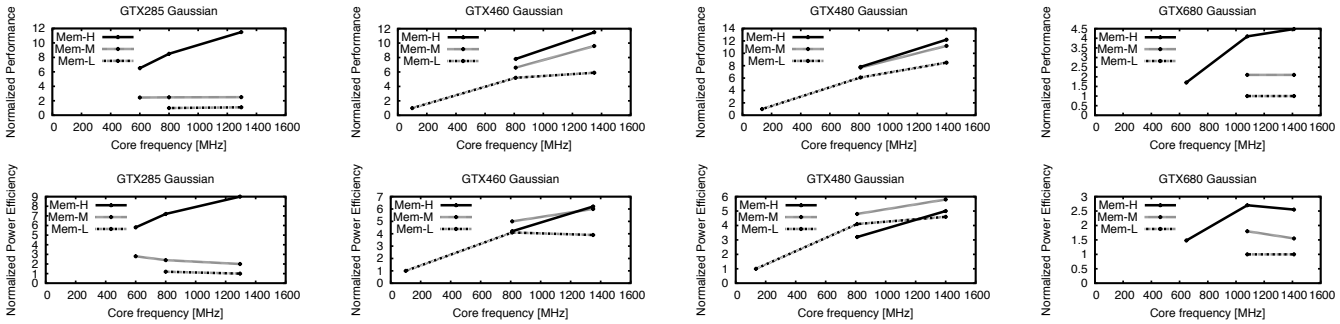

Fig. 3. Performance and power efficiency of `Gaussian`.

pair. Because (H-H) is the default setting, we highlight other configurations by the bold font in the table. It is very interesting to see that (H-H) achieves the best power efficiency for almost all benchmarks for GTX 285. However, we can see that the best pair becomes more diverse as the generation proceeds, and surprisingly for GTX 680, the best power efficiency for all the benchmarks are achieved besides the default configuration. This characteristic explains that processor and memory voltage and frequency scaling is a useful approach to minimizing energy of GPU-accelerated systems in the current state of the art.

Fig. 4 shows the power efficiency improvement (i.e., energy reduction) that can be achieved by selecting the best configuration over the default (H-H) configuration. For GTX 285, only four out of 17 benchmarks need different configurations beside (H-H) for the best power efficiency, and only 0.8% improvement can be achieved by optimal selections. For Fermi GPUs, we can perform much better than GTX 285 where 12.3% and 12.1% improvements are achieved for GTX 460 and GTX 480, respectively. When it comes to the latest GTX 680, the efficiency improvement reaches 24.4% on average, where the improvements for six benchmarks (`LavaMD`, `Pathfinder`, `SRAD`, `sad`, `sgemm` and `spmv`) exceed 40%. To summarize, we have shown that energy optimization of GPUs is very challenging, given the factors of hardware design knobs which are core and memory frequency pairs and also the characteristics of the workloads.

IV. Statistical Modeling

A. Model Construction:

We now provide statistical power and performance modeling of GPU-accelerated systems based on the characterization presented in Section III. The goal of this modeling is to predict power and performance behavior across multiple generations of architectures with various processing core and memory frequency pairs.

TABLE IV
THE BEST FREQUENCY PAIRS FOR POWER EFFICIENCY.

Benchmarks	GTX 285	GTX 460	GTX 480	GTX 680
Rodinia	(Core frequency-Memory frequency)			
Backprop	**(H-L)**	**(H-L)**	**(H-L)**	**(M-L)**
BFS	**(M-H)**	(H-H)	(H-H)	**(M-H)**
CFD	(H-H)	(H-H)	(H-H)	**(M-M)**
Gaussian	(H-H)	(H-H)	**(H-M)**	**(M-H)**
Heartwall	(H-H)	**(H-M)**	**(H-M)**	**(L-H)**
Hotspot	(H-H)	**(H-L)**	**(H-L)**	**(M-L)**
Kmeans	(H-H)	(H-H)	**(M-M)**	**(M-M)**
LavaMD	(H-H)	**(H-L)**	**(H-M)**	**(H-L)**
Leukocyte	(H-H)	**(H-L)**	**(H-L)**	**(H-M)**
LUD	(H-H)	**(H-M)**	**(H-M)**	**(L-H)**
MUMmerGPU	(H-H)	(H-H)	(H-H)	**(M-H)**
NN	(H-H)	**(H-M)**	**(H-L)**	**(H-L)**
NW	(H-H)	**(H-M)**	**(H-M)**	**(L-H)**
Particlefilter	**(H-M)**	**(H-L)**	**(H-L)**	**(H-L)**
Pathfinder	**(H-M)**	**(H-M)**	**(H-M)**	**(H-M)**
SRAD	(H-H)	(H-H)	(H-H)	**(L-H)**
Streamcluster	(H-H)	(H-H)	(H-H)	**(M-H)**
Parboil	(Core frequency-Memory frequency)			
cutcp	(H-H)	**(H-M)**	**(H-L)**	(H-H)
histo	(H-H)	(H-H)	**(M-M)**	(H-H)
lbm	(H-H)	(H-H)	**(M-H)**	**(M-H)**
mri-gridding	**(M-M)**	**(H-L)**	**(M-M)**	**(M-M)**
mri-q	(H-H)	**(H-L)**	**(H-L)**	**(M-H)**
sad	(H-H)	(H-H)	(H-H)	**(M-M)**
sgemm	(H-H)	**(H-M)**	**(M-M)**	**(H-M)**
spmv	(H-H)	**(H-L)**	**(H-L)**	**(M-H)**
stencil	(H-H)	(H-H)	(H-H)	(H-H)
tpacf	**(H-L)**	**(H-M)**	**(H-M)**	**(H-M)**
CUDA SDK	(Core frequency-Memory frequency)			
binomialOptions	**(H-L)**	**(H-L)**	(H-H)	**(M-M)**
BlackScholes	(H-H)	(H-H)	(H-H)	**(M-H)**
concurrentKernels	**(L-M)**	**(L-L)**	**(L-L)**	**(M-M)**
histogram256	(H-H)	**(M-M)**	**(H-M)**	**(M-M)**
histogram64	(H-H)	**(H-M)**	**(M-M)**	**(H-M)**
MersenneTwister	**(L-M)**	(H-H)	(H-H)	**(M-H)**

Our statistics-based approach uses *multiple linear regression* where power and performance are used as dependent variables while the statistical data obtained via performance counters are used as independent variables. We leverage the results of Section III in coordination with the performance counter information obtained by the CUDA Profiler v2.01.* Different from any other prior work, we propose a unified model which incorporates the frequencies of both core and memory into the prediction equation such that a single model for each GPU allows to predict power and performance for any given frequency pair.

We construct models of power and performance individually for all the evaluated GPUs where we can examine whether the accuracy and contributing performance counters differ among GPUs. Note that the types and the number of performance counters depend on each GPU architecture: 32 counters for GTX 285, 74 counters for GTX 460 and GTX 480, and 108 counters for GTX 680. All the benchmark programs shown in TABLE II except for three (`mummergpu`, `backprop` and `pathfinder`) from Rodinia and one (`bfs`) from Parboil, which failed to be analyzed by the CUDA Profiler, are used as modeling samples. We finally obtain 114 samples in total by changing the size of inputs for each benchmark. We use the forward selection method to find an "optimal" model that maximizes the adjusted coefficient of determination ($\bar{R}^2$) by allowing at most 10 independent variables to be used. We have tried using 15 and 20 variables instead of 10 and confirmed that the $\bar{R}^2$ values almost does not improve when increasing the number of independent variables beyond 10. The details of the proposed power and performance models are described in the following.

*We did not use emulation techniques (e.g., GPU Ocelot [4]) although they provide different statistics from performance counters; it requires too much time to complete all the benchmarks we have evaluated. Recent advances in instrumentation techniques (e.g., Lynx [5]) can overcome this issue and using them for analysis is left for future work.

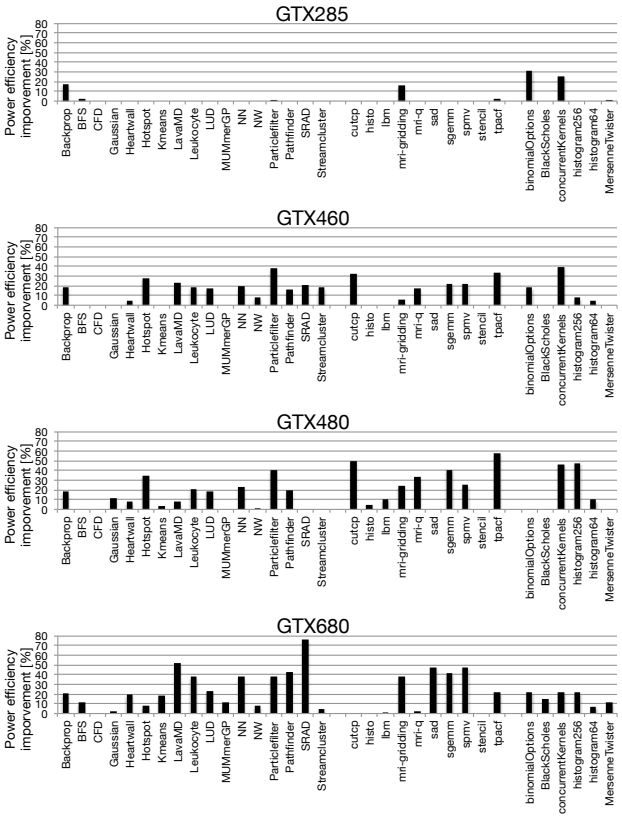

Fig. 4. Power efficiency improvement with the best configuration.

Power Modeling: Previous study has shown that reasonable estimation is possible using multiple linear regression for a single frequency pair (core and memory) [18]. We propose a unified model which demonstrates high estimation accuracy for any frequency pairs. In order to incorporate the information of operating frequency into a single equation, we need to consider that an access to a certain component consumes different amount of energy when operating at different frequency levels. We account for this phenomenon by categorizing the performance counters into two groups: *core-event* or *memory-event*, whose power consumption associated with the event depends on the core or memory frequency, respectively.† Each value of performance counter grouped as core-event is multiplied by the core frequency (i.e., the faster the frequency, the more power consumed for each access), and the same is

†We do not show what counters are classified into which group because of space limitations. In general, core-events are the events which happen within the core where memory-events are un-core events such as memory accesses.

TABLE V
$\bar{R}^2$ OF THE POWER MODEL.

GTX 285	GTX 460	GTX 480	GTX 680
0.30	0.59	0.70	0.18

TABLE VI
$\bar{R}^2$ OF THE PERFORMANCE MODEL.

GTX 285	GTX 460	GTX 480	GTX 680
0.91	0.90	0.94	0.91

done for the memory-event. Therefore, the statistical model for predicting power consumption is formed as:

$$power = \sum_{i=1}^{Nc} x_i c_i \cdot corefreq + \sum_{j=1}^{Nm} y_j m_j \cdot memfreq + z \quad (1)$$

where $power$ denotes the dependent variable (power consumption), c_i $(1 \leq i \leq Nc)$ and m_j $(1 \leq j \leq Nm)$ are the per-second scale performance counter values (in order to predict the average W of the program [18]), x_i, y_i and z are the model coefficients.

Performance Modeling: On the performance prediction side the model also incorporates the frequencies like the power model but in a slightly different manner. Categorization of the events are the same, however, each value of performance counter grouped as core-event is divided (not multiplied) by the core frequency (i.e., the faster the frequency, the shorter the latency associated with the event), and the same is done for the memory-event. After all, the performance prediction model is formed as:

$$exectime = \sum_{i=1}^{Nc} x_i \frac{c_i}{corefreq} + \sum_{j=1}^{Nm} y_j \frac{m_j}{memfreq} + z \quad (2)$$

where $exectime$ denotes the dependent variable (total execution time), c_i $(1 \leq i \leq Nc)$ and m_j $(1 \leq j \leq Nm)$ are the performance counter values (here we use the values from the total run which is similar in spirit with the work by Hong et al. [7, 8]) grouped as core-event and memory-event, respectively, $corefreq$ is the core frequency, $memfreq$ is the memory frequency, x_i, y_j and z are the model coefficients.

Note that in contrast to previous studies [1, 7, 23] which required expert knowledge of GPU architectures and built complicated performance models, we attempt to build a statistical model with only a minimum knowledge of workload characteristics. One contribution of this study is that it shows the limitation of multiple linear regression when applied for performance modeling.

B. Model Evaluation:

Tables V and VI show the $\bar{R}^2$ values of the obtained power and performance models for each GPU. We can see that the variations of the $\bar{R}^2$ values for the power model is huge (0.18 to 0.70) where that of the performance model is small with significant high values (0.90 to 0.94). It is somewhat counter intuitive to see that the performance model achieves a much higher $\bar{R}^2$ values than the power model because previous study has shown that the power consumption can be reasonably predicted with multiple linear regression [18]. However, the obtained result is not different from theirs which means that the power prediction model can be used in a real system with small errors as we will show in the following paragraph.

TABLE VII
AVERAGE PREDICTION ERROR OF THE POWER MODEL.

	GTX 285	GTX 460	GTX 480	GTX 680
Error[%]	15.0	14.0	18.2	23.5
Error[W]	20.1	15.2	24.4	23.7

TABLE VIII
AVERAGE PREDICTION ERROR OF THE PERFORMANCE MODEL.

	GTX 285	GTX 460	GTX 480	GTX 680
Error[%]	67.9	47.6	39.3	33.5

Fig. 5 and Fig. 6 exhibit errors in prediction using our statistical power and performance models for each GPU. Tables VII and VIII complement these figures by showing the average value of the absolute errors in percentage for each model. TABLE VII also shows that of errors in Watts. As discussed earlier, even though the values of $\bar{R}^2$ for the power model is inferior to that of the performance model, the overall trend of results implies that the power model is accurate enough to cap errors in prediction. This is attributed to the fact that the variations of power consumption are limited within 100W for each GPU in contract to the execution time, which varies from hundreds of milliseconds to tens of seconds. Mathematically a small variation of the target estimate leads to a small value of the adjusted coefficient of determination. Thus, a small value of $\bar{R}^2$ does not necessarily mean that the model is inaccurate. In fact, the errors in prediction generated from our power model are no greater than 20.1W, 15.2W, 24.4W and 23.7W for GTX 285, 460, 480 and 680, respectively, though the model seems somewhat inaccurate at the first glance according to the values of $\bar{R}^2$.

However, it can be observed that the prediction accuracy is lower for newer generation GPUs. Building a more sophisticated model to improve the accuracy is left for future work. In contrast to the power model, errors in prediction generated from our performance model are more significant as shown in Fig. 6 and TABLE VIII, despite of its very high values of $\bar{R}^2$. The x-axis shows benchmark sorted independently for each GPU. In fact, this result is expected in a sense that the variations of the execution time widely ranges from milliseconds to tens of seconds and such a larger variation of the target estimate leads to a smaller value of the adjusted coefficient of determination. However, the result is still encouraging in the following: (i) even when using a simple linear model presented in this paper, the values of $\bar{R}^2$ can reach greater than 0.90 for all GPUs, and (ii) it can be clearly seen that the errors in prediction are becoming smaller as a generation of the GPU architecture proceeds. We believe that this is due to an increased number of available performance counters in recent architectures and the enhanced microarchitecture can also remove unpredictable behaviors present in old GPUs. The above discussion leads to a conclusion that the performance of future lines of GPUs will be more predictable even under simple statistical models while the power model needs further

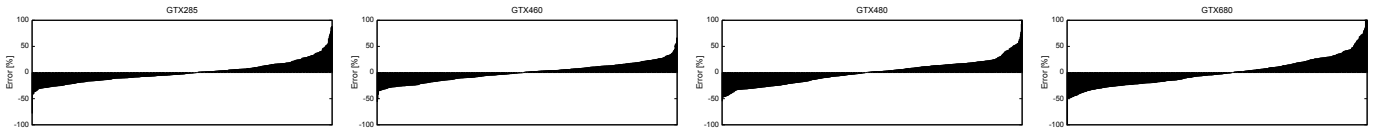

Fig. 5. Errors in prediction of the power model (by distribution over all benchmarks).

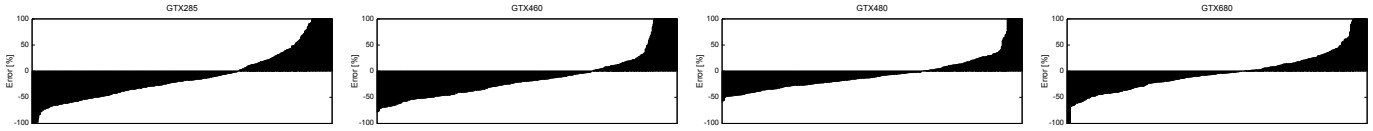

Fig. 6. Errors in prediction of the performance model (by distribution over all benchmarks).

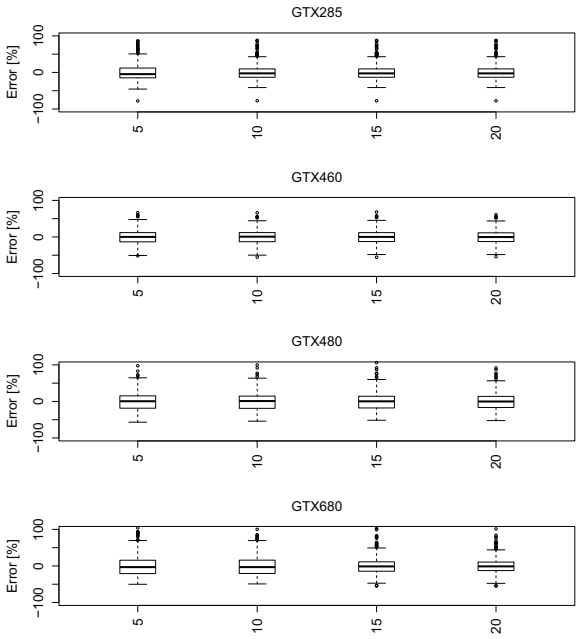

Fig. 7. Impact of explanatory variables on the power model.

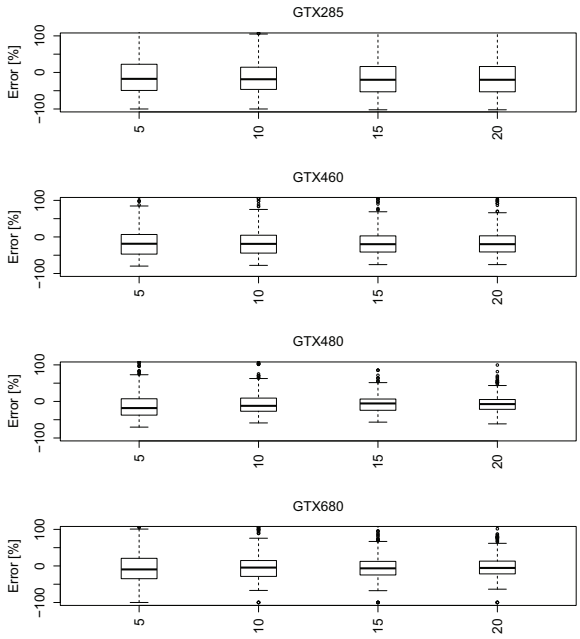

Fig. 8. Impact of explanatory variables on the performance model.

improvement as the hardware becomes more power efficient. This paper contributes significantly to this novel finding.

Overall, we find that even simplified statistical models are able to predict power and performance of advanced GPU-accelerated systems (i.e., NVIDIA's Kepler) within errors of 20% to 30% for any set of voltage and frequency pair and for any workload. In fact, more than half of the workloads in Fig. 5 and Fig. 6 exhibit prediction errors less than 20% for power and performance on all the evaluated GPUs. This means that even simplified statistics are very powerful to predict power and performance of GPU-accelerated systems, though depending on the workload. Our future work is to validate the proposed power performance models by targeting multiple GPU microarchitectures as NVIDIA's Kepler and AMD's Radeon.

As aforementioned, we use 10 explanatory variables to evaluate our power and performance models. For reference, we also evaluate our models using 5 to 20 explanatory variables as shown in Fig. 7 and Fig. 8. The x-axis shows the number of explanatory variables. We can see that our configuration of using 10 variables gives reasonable prediction accuracy for both power and performance prediction.

Fig. 9 and Fig. 10 show the box and whisker plots of the error distributions for the proposed model on the right hand side along with models constructed for each frequency pair (e.g., Core-High_Mem-High) in order to understand the effectiveness of the unified model. In general, the slight difference between each model versus our unified model show that even though both the power and performance prediction models become more accurate when optimized for each frequency pair, the proposed model exhibits its advantage of a simple and a uniform model. As discussed earlier, Fig. 9 shows that the power model is highly accurate when tied to a single model, which is also confirmed by the previous work by Nagasaka et al. [18]. As TABLE VIII showed, it can be seen that the accuracy for the performance model improves as the GPU generation proceeds. We can see that the accuracy comes from not some specific frequency pair but is the overall trend of each

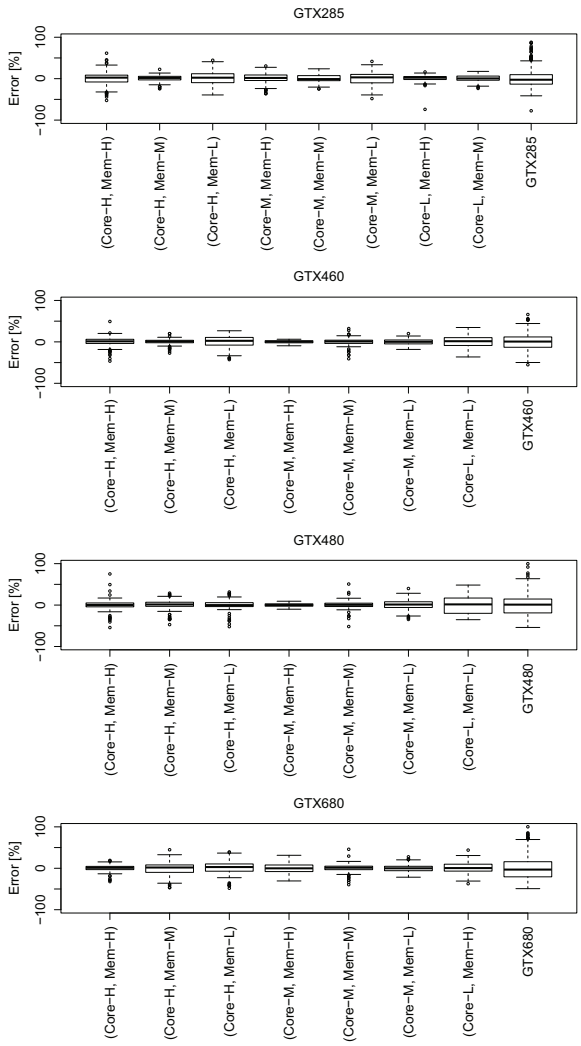

Fig. 9. Impact of GPU clocks on the power model.

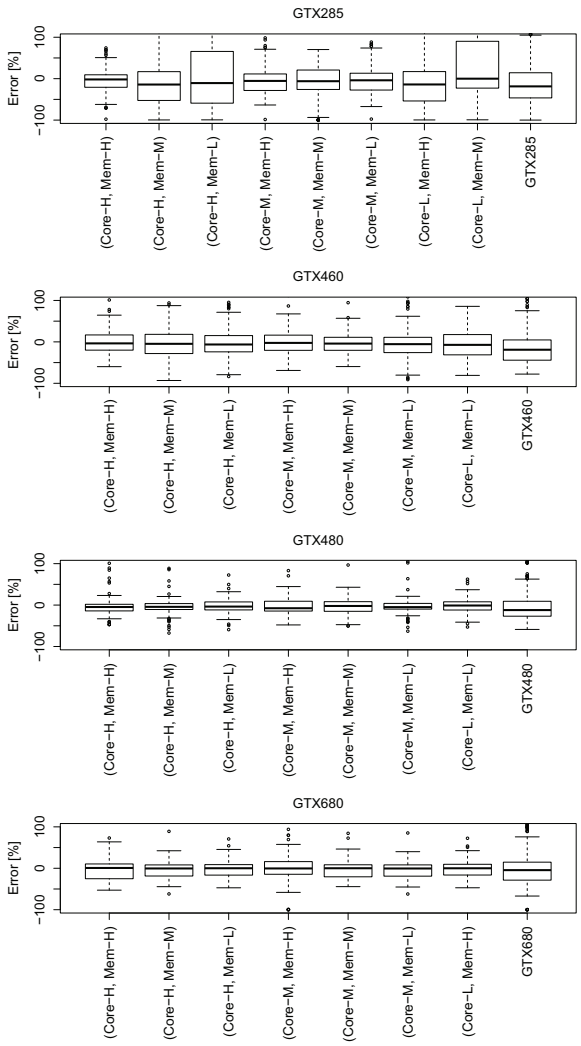

Fig. 10. Impact of GPU clocks on the performance model.

GPU. Surprisingly, some models show very wide variations even when optimized for each configuration. It can be seen that the unified model can incorporate them to perform a reasonable prediction. An important finding is that the applied simple multiple linear regression is effective in constructing accurate models even by unifying all the frequency pairs into a single model for all the GPU architectures. This fact suggests that statistical modeling continues as a promising approach which does not require in-depth understanding (which is sometimes impossible because of the black-box nature of GPUs) of the architecture itself (which was required in some of the previous studies [7, 8]).

Finally, Fig. 11 shows breakdown of the impact of influence from selected explanatory variables. Detailed explanations of these explanatory variables are outside the scope of this paper, though this breakdown evidence that there are at most 10 to 15 variables that really influence power and performance of GPU-accelerated systems. We believe that it is fairly realistic to select 10 to 15 variables at runtime to dynamically predict power and performance of GPU-accelerated systems.

V. Related Work

Hong et al. developed an integrated power and performance model for NVIDIA Tesla GPUs [7, 8]. They analyze PTX expression code to derive the number of instructions and memory accesses, forming the model based on these pieces of information. This off-line PTX analysis approach is highly useful to predict the power and performance of GPUs. One of the shortcomings of this work is that the resultant model is specific to a GTX 280 GPU. We tried to apply the model to a GTX 285 GPU, which is based on the same Tesla architecture, but it was very time-consuming to tune the parameters of the model for the new GPU. Given that the Fermi and the Kepler architectures have significant leap over the Tesla architecture, a detailed analysis in PTX expression code for each different GPU based on these architectures may be more time-consuming. Our statistical modeling approach provides a unified form for all GPUs, which is a significant advantage in modeling new GPUs.

Baghsorkhi et al. compensated for prior work [7] by revisiting the program dependence graph [1]. This is an adap-

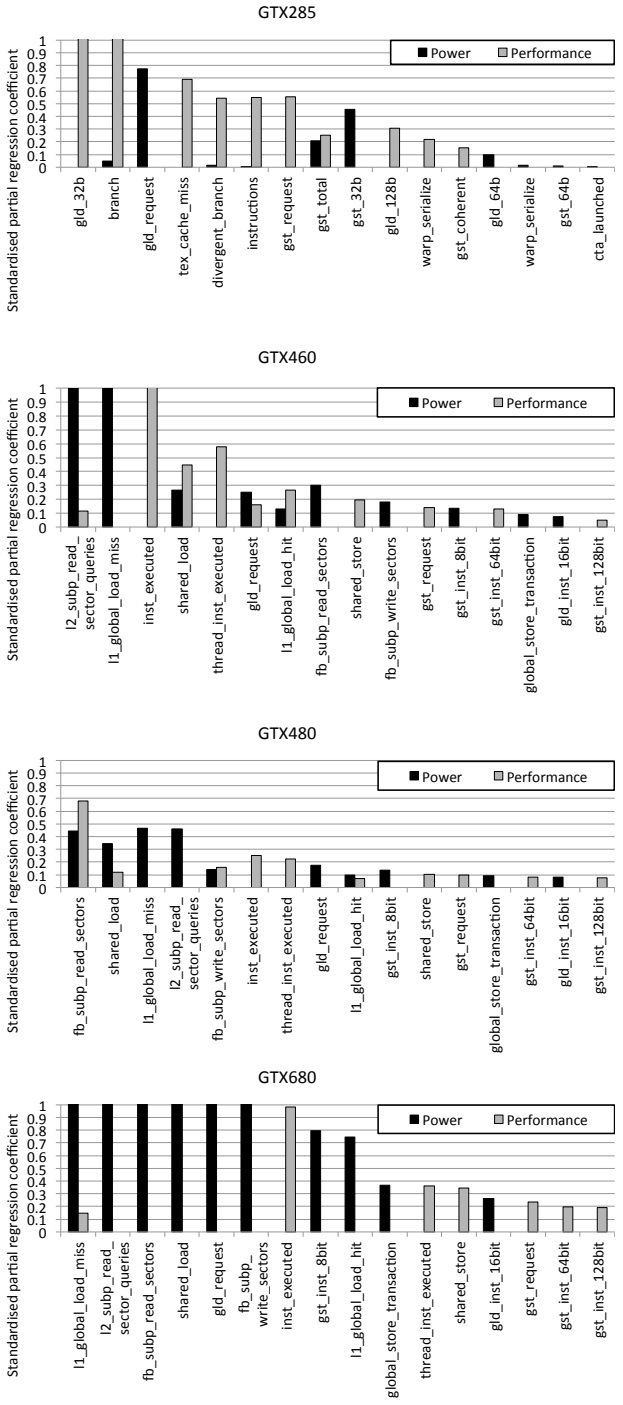

Fig. 11. Selected explanatory variables and their impact of influence on power and performance.

tive compiler-based approach, capturing the control flow of computing kernels or memory bank conflicts that are not considered in the prior work. They showed that the execution times of simple matrix operations can be precisely predicted on an NVIDIA Tesla GPU (GeForce 8800) under their adaptive performance model.

Lee et al. presented a methodology to use DVFS algorithms for NVIDIA Tesla GPUs, aiming to maximize performance under given power constraints [14]. A strong limitation of their work is that the evaluation of power consumption is limited to a conceptual model. Our evaluation and modeling are based on real hardware. This is a useful evidence when translating theory into practice.

Nagasaka et al. conjectured a power model of NVIDIA Tesla GPU using statistics and the hardware performance counters [18]. The contribution of our work over theirs is a generalization of their result. Our model conjectures both the power and the performance of GPUs across different architectures, while their work is focused on the power behavior of a GTX 285 GPU. This extension comes from our significant engineering effort where we figured out how to control the voltage and frequency of NVIDIA GPUs by analyzing the format and semantics of their BIOS image. Given that NVIDIA GPUs are the current state of the art but are black-box devices, our open methodology for voltage and frequency scaling of NVIDIA GPUs is a useful contribution aside scientific findings.

Jiao et al. evaluated the power and performance of an NVIDIA Tesla GPU (GTX 280) for compute-intensive and memory-intensive workload applications [12]. According to their discussion, energy consumption could often be reduced by lowering the processing core frequency when workload is memory-intensive. In this paper, we have found that the processing core and memory frequencies are in a complicated causal relation when executing more realistic workload. While it is often effective to lower the processing core frequency for memory-intensive workload or vice versa, our evaluation demonstrates that this is not always the case, depending on both the workload and the GPU.

Ma et al. investigated a heterogeneous energy management framework using an NVIDIA Tesla GPU (GeForce 8800) [16]. They demonstrated that (i) a workload distribution between the CPU and the GPU and (ii) coordinated processing core and memory scaling for the GPU are important control knobs for energy optimization. Their contribution encourages this paper in that both the processing core and memory frequencies of the GPU are considered. However, this paper is distinguished in that we performed our evaluation and modeling across multiple generations of GPUs to address integrated power and performance issues of GPU-accelerated systems.

The contributions of the above related work are limited to an individual GPU based on the ancient Tesla architecture. However, the current state of the art of NVIDIA GPUs is mostly based on the Kepler and Fermi architectures whose designs have been highly renovated over the Tesla architecture. It is questionable if the models or power and performance management schemes presented by the previous work still hold for these new architectures. Our contribution is more generalized across multiple generations of the GPU architecture.

Sim et al. deepened the preceding analytic performance model [7] by introducing (i) a new way to identify performance bottlenecks and (ii) new metrics for more flexible performance prediction [23]. In theory, this extended model is applicable for any GPU based on the NVIDIA Fermi architecture. In practice, however, the performance behavior is not consistent among different GPUs even for the same architecture as demonstrated in this paper. It is worth exploring

if their model is applicable for different GPUs or not. Note that they focus only on performance (and not power consumption).

There are several other prior work conducted using AMD GPUs. Zhang et al. analyzed the power and performance of a Radeon HD 5870 GPU using a random forest method with the profile counter information [27]. They concluded that activating a fewer number of ALUs can reduce power consumption. A shortcoming of this approach is that the performance loss does not pay for energy saving. Voltage and frequency scaling is more efficient as demonstrated in this paper. This is attributed to the fact that they restrict their approach to software management. Liu et al. developed a power-efficient task mapping scheme for real-time applications on heterogeneous platforms using a Radeon HD 5770 GPU [15]. What could be complemented for their work is an integration of memory performance scaling. Therefore the contribution of this paper could further extend and generalize their result.

All in all, we are not aware of any previous study that successfully unifies operating frequency into a single model. Hence previous models must be constructed for each different operating frequency level which forces the system designers to consider multiple instances of the model even for a single GPU when applying the model for a real system. On the other hand, our statistical modeling approach allows the model to contain operating frequency as one of the model parameters. This unified model would be a strong basis for the dynamic runtime management of power and performance for GPU-accelerated systems.

VI. Conclusions

In this paper, we have presented power and performance characterization and modeling of GPU-accelerated systems. We selected four different NVIDIA GPUs from three generations of architectures in order to demonstrate generality of our contribution. One of our notable findings is that a pair of the GPU processing core and memory clocks for minimized energy is becoming more and more diversified as a generation of the GPU architecture proceeds. This evidence encourages future work on the management of power and performance for GPU-accelerated systems to benefit from dynamic voltage and frequency scaling. We also demonstrated that our statistical power and performance models are reliable enough to predict power and performance across multiple generations of architectures.

Acknowledgement

This research was supported in part by Japan Science and Technology Agency (JST), CREST.

References

[1] S. S. Baghsorkhi, M. Delahaye, S. J. Patel, W. D. Gropp, and W.-m. W. Hwu, "An adaptive performance modeling tool for GPU architectures," in *Proc. of the ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2010.

[2] S. Che, J. Sheaffer, M. Boyer, L. Szafaryn, L. Wang, and K. Skadron, "A characterization of the Rodinia benchmark suite with comparison to contemporary CMP workloads," in *Proc. of the IEEE International Symposium on Workload Characterization*, 2010.

[3] Y. E. Corporation, "WT1600 digital power meter," 2012, http://tmi.yokogawa.com/discontinued-products/digital-power-analyzers/digital-power-analyzers/wt1600-digital-power-meter/.

[4] G. Diamos, A. Kerr, S. Yalamanchili, and N. Clark, "Ocelot: a dynamic optimization framework for bulk-synchronous applications in heterogeneous systems," in *Proc. of the International Conference on Parallel Architectures and Compilation Techniques*, 2010.

[5] N. Farooqui, A. Kerr, G. Eisenhauer, K. Schwan, and S. Yalamanchili, "Lynx: A dynamic instrumentation system for data-parallel applications on GPGPU architectures," in *Proc. of the IEEE International Symposium on Performance Analysis of Systems and Software*, 2012.

[6] S. Hand, K. Jang, K. Park, and S. Moon, "PacketShader: a GPU-accelerated software router," in *Proc. of ACM SIGCOMM*, 2010.

[7] S. Hong and H. Kim, "An analytical model for a GPU architecture with memory-level and thread-level parallelism awareness," in *Proc. of the ACM/IEEE International Symposium on Computer Architecture*, 2009.

[8] S. Hong and H. Kim, "An integrated GPU power and performance model," in *Proc. of the ACM/IEEE International Symposium on Computer Architecture*, 2010.

[9] C. Isci, A. Buyuktosunoglu, C.-Y. Cher, P. Bose, and M. Martonosi, "An analysis of efficient multi-core global power management policies: maximizing performance for a given power budget," in *Proc. of the IEEE/ACM International Symposium on Microarchitecture*, 2006.

[10] C. Isci, G. Contreras, and M. Martonosi, "Live, runtime phase monitoring and prediction on real systems with application to dynamic power management," in *Proc. of the IEEE/ACM International Symposium on Microarchitecture*, 2006.

[11] K. Jang, S. Han, S. Han, S. Moon, and K. Park, "SSLShader: cheap SSL acceleration with commodity processors," in *Proc. of the USENIX Conference on Networked Systems Design and Implementation*, 2011.

[12] Y. Jiao, H. Lin, P. Balaji, and W. Feng, "Power and performance characterization of computational kernels on the GPU," in *Proc. of the IEEE/ACM International Conference on Green Computing and Communications*, 2010.

[13] S. Kato, M. McThrow, C. Maltzahn, and S. Brandt, "Gdev: first-class GPU resource management in the operating system," in *Proc. of the USENIX Annual Technical Conference*, 2012.

[14] J. Lee, V. Sathisha, M. Schulte, K. Compton, and N. S. Kim, "Improving throughput of power-constrained GPUs using dynamic voltage/frequency and core scaling," in *Proc. of the International Conference on Parallel Architectures and Compilation Techniques*, 2011.

[15] C. Liu, J. Li, W. Huang, J. Rubio, E. Speight, and X. Lin, "Power-efficient time-sensitive mapping in heterogeneous systems," in *Proc. of the International Conference on Parallel Architectures and Compilation Techniques*, 2012.

[16] K. Ma and X. Wang, "GreenGPU: a holistic approach to energy efficiency in GPU-CPU heterogeneous architectures," in *Proc. of the International Conference on Parallel Processing*, 2012.

[17] M. McNaughton, C. Urmson, J. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Proc. of the IEE International Conference on Robotics and Automation*, 2011.

[18] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, and S. Matsuoka, "Statistical power modeling of GPU kernels using performance counters," in *Proc. of the Green Computing Conference*, 2010.

[19] NVIDIA, "NVIDIA's next generation CUDA computer architecture: Fermi," 2009, http://www.nvidia.com/content/PDF/fermi_white_papers/NVIDIA_Fermi_Compute_Architecture_Whitepaper.pdf.

[20] NVIDIA, "CUDA TOOLKIT 4.2," 2012, http://developer.nvidia.com/cuda/cuda-downloads.

[21] NVIDIA, "NVIDIA's next generation CUDA computer architecture: Kepler GK110," 2012, http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf.

[22] NVIDIA, "CUDA Documents," 2013, http://docs.nvidia.com/cuda/.

[23] J. Sim, A. Dasgupta, H. Kim, and R. Vuduc, "A performance analysis framework for identifying potential benefits in GPGPU applications," in *Proc. of the ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2012.

[24] J. Stratton, C. Rodrigues, I.-J. Sung, N. Obeid, L.-W. Chang, N. Anssari, G. Liu, and W.-M. Hwu, "Parboil: a revised benchmark suite for scientific and commercial throughput computing," IMPACT-12-01, University of Illinois at Urbana-Champaign, Tech. Rep., 2012.

[25] W. Sun, R. Ricci, and M. Curry, "GPUstore: harnessing GPU computing for storage systems in the OS kernel," in *Proc. of Annual International Systems and Storage Conference*, 2012.

[26] TOP500 Supercomputing Site, 2012, http://www.top500.org/.

[27] Y. Zhang, Y. Hu, B. Li, and L. Peng, "Performance and power analysis of ATI GPU: a statistical approach," in *Proc. of the IEEE International Conference on Networking, Architecture, and Storage*, 2011.